AN ONTOLOGY OF PHYSICAL ACTION

By

Ernest Davis

June 1984

Technical Report #123

An Ontology of Physical Actions

Ernest Davis

ABSTRACT

This paper presents an ontology for reasoning about the use of rigid objects as tools. We define the concepts of scene, history, constraint, sensor, action, and feasibility in extensional terms, and discuss their application to our system. Also, we present formalizations of several inituitive physical laws.

May 23, 1984

c.1

An Ontology of Physical Actions

Ernest Davis

1. Introduction

This paper presents a formal ontology and language for representing knowledge about actions involving physical objects. Our ultimate aim is to develop a knowledge base manager to perform the inferences required a robot who uses tools. The following questions might be typical of those to be presented to our knowledge base manager.

a) How can the robot hang a hanger on a hook?

b) Will a screen with a large hole keep out flies?

c) Which of the available tables will hold drinks at a convenient level next to the sofa?

d) How can a one-handed robot carry four books, a pencil, and a pencil sharpener from one place to another if the available tools are a ladder, a box, and a wheel?

e) What kind of hammer claw is needed to remove a tack hammered into the wall one millimeter above the floor?

f) How can a robot with only sonar find his way to the door without scraping the walls? Suppose he has equipped with felt-covered bumpers, so that he is allowed to scrape the walls?

The basic components of our problems are a robot, who consists of a body and sensors; a starting scene, with specific objects in specific places; an objective to be achieved; a number of physical constraints (e.g. the robot cannot go through walls; the robot cannot disconnect his own hand); a number of imposed constraints (the robot is not allowed to rub against the walls); and an action which the robot is to carry out. In general, a problem specifies some of these, and poses the problem of finding the rest. For example, in (a), (d), and (f) we are required to find appropriate

actions. In (b) we are asked whether a given object will serve a given objective. In (c) and (e) we are asked to find or design a particular object for a particular objective.

This is a study in naive physics, similar to [Bundy, 83], [Forbus, 83], or [Kuipers, 82]. However, the above problems require much deeper spatial representation and reasoning than is found in the naive physics literature. We will study the interface between spatial and physical representations in detail in subsequent papers. This paper presents only the ontological primitives needed for the domain, in the style of [Hayes, 79], [McDermott, 82], or [Moore, 80].

We begin with the basic ontology of modelling physical space in terms of R^3 and time in terms of R, and then we define these more complex concepts in terms of these using the language of sets and functions. Simultaneously, we build up the primitives of a first order language, and indicate how we could use the language to express the knowledge which we wish to incorporate into our system. For the time being we will slough over the essential question of what kinds of geometric primitives are needed and simply help ourselves to any geometric primitive in the mathematical literature. Future papers will consider the problem of forming a geometric representation for this system.

2. Scope

We impose three basic restrictions on the scope of our problem. Firstly, we consider only rigid objects, not flexible or fluid objects. We describe a jointed objects as a set of rigid objects whose positions are restricted by physical constraints. Secondly, we consider only actions whose effect is essentially independent of the speed at which they are performed. Thus, pushing, carrying, or blocking actions would be included but hitting a baseball or intercepting a missile would not. Thirdly, we consider only fixed numbers of objects, not sets of objects of undetermined cardinality. These restrictions very much simplify the problem, while still leaving a very large class of cases. Most of the situations which have been considered in the study of robotics fall into these classes.

Given the restriction to rigid objects, why do we not simply use Newtonian physics, with its

gestions. In (b) we are asked to relate [illegible] of a set of objects [illegible] we are asked to find or design a [illegible] sequence [illegible] objective.

This is a study in [illegible] (Barlow, [3]), [illegible] ever, the above problems require [illegible] in the naive physics [illegible] and geometrical [illegible] done in detail in [illegible] logical primitives [illegible] the domain, in the [illegible]

We begin with the [illegible] terms of R, and [illegible] of sets and functions [illegible] indicate how we could [illegible] into our system. [illegible] geometric primitives [illegible] mathematical literature [illegible] tation for the [illegible]

2. Shape

We begin [illegible] rigid objects, [illegible] whose position [illegible] effect is essentially [illegible] bag, or floating [illegible] ant. Thirdly, [illegible] rigidity. There [illegible] ment. More [illegible] dition.

There [illegible]

great power? The truth is that an analysis in terms of forces, energy, and momentum is often both unnecessary and insufficient. For example, if we wish to predict what happens to books in a box if a box is moved, it is impossible to say exactly what horizontal forces will be applied to the books when. Nor is it necessary; we know that the books will remain in the box. For another example, if a book rests on a table, Newton's laws do determine that the book will stand still, but the derivation is not easy, and a much simpler analysis of the situation will suffice.

We choose, instead, to go to the opposite extreme and consider only cases which can be analyzed without any laws depending on time measurement. The primary law which we will depend on is that two hard objects cannot overlap. In this limited view, we analyze "A pushing B" as follows. For some reason A is "trying" to go forward. However, B is in front of A, so A cannot simply move forward, leaving the rest of the world unchanged. Rather, if at all possible, the world accomodates itself to A's attempt while preserving the constraint. This is done by B moving forward. If B cannot move forward (say it is glued to the ground) then A's attempt is not successful. The remainder of the paper is basically the development of a system in which the above analysis can be expressed.

Two further restrictions on the scope of this enterprise should be noted. The general problem of moving rigid objects around rigid barriers is very difficult. (See [Schwartz and Sharir, 81], [Lozano-Perez, 81] and [Brooks, 82].) For our purposes, we are not looking for exact or complete algorithms, merely for quick usable heuristics. The other restriction is that we wish to avoid problems requiring sophisticated planning; in particular, those raised by interacting goals. (See, for example, [McDermott, to appear], [Wilensky, 80], or [Sacerdoti, 75].)

Many of the definitions and axioms below are formulated in predicate calculus mixed with set theoretic notation. These should be self-explanatory. Let me merely make the following remarks as to notation:

a) Binary boolean operators (and, or, implies, iff) and the primitive relations "$=$", "$\equiv$", "$\neq$" "$\bigcup$", "$\bigcap$" and "ϵ" (element) are infix. All other predicates, including "subset", are prefix.

b) Functions and relations are indicated in the normal mathematical style $f(x)$ rather than in the LISP style $(f\ x)$. In cases where f is itself a complex expression, we will either write "apply(f,x)" or invent a notation suggestive of the type of f.

c) Indentation is used extensively for grouping. Where this is unclear, I have added parentheses and brackets. Curly brackets "{}" are used exclusively for set notation.

d) I used a typed logic, where the type of variable is indicated by the first letter. T is used for times; S, for scenes; H for histories; A, for actions. Also, I use quantification in sets. "forall $(X \in S) P(X)$" means "forall (X) [$(X \in S)$ implies $P(X)$]"

e) Any variable which is not explicitly quantified is assumed to be universally quantified.

f) Where predicate calculus would be complicated and unhelpful, I have stuck to English.

3. Basics

We begin with a number of basic definitions.

The building blocks of our system are subsets of three-dimensional space (R^3). Since there is no standard term for these, we will call them *bodies*.

Definition 3-1:

A *body* is a closed, bounded subset of R^3.

The closedness condition is purely formal, to simplify some of the axioms and definitions below. Boundedness and connectedness are physically plausible, and simplify the exposition.

We assume the existence of a set of distinguishable objects. We might be tempted to identify an object with the body it occupies, or, since objects can move around, with the set of all bodies it can occupy in different positions. This is not adequate, however, since it does not distinguish between different congruent embeddings of a symmetric objects. For example, we would not be able to describe a cylinder spinning around its axis. Rather we define the shape of an object O in terms of a *standard position*, written *stand*(O), which is a body. Then we describe all other positions of the object as rigid mappings which take stand (O) into some other congruent body.

Axiom 3-1:

For each object O, stand (O) is a body.

Definition 3-2:

A *mapping* is a sense-preserving (i.e. no reflections) rigid transformation. Given a mapping M and a point set P, we write "image(M,P)" as alternate notation for $M(P)$.

We now define a scene to be an arrangement of objects. The positions of an object in a scene is defined in terms of a mapping from its standard position.

Definition 3-3:

A *scene* is a function from a set of objects onto mappings (1 mapping per object). Given a set of objects, U, we write "scene(S,U)" to mean that S is a scene with domain U.

Some useful notation with scenes:

Definition 3-4:

Given a set of objects U, a scene S with domain U, an object O in U, and a subset P of stand(O),

a) scenemapping(S,O)$\equiv S(O)$, the mapping which locates O in S.

b) sceneplace (P,O,S)$\equiv$ image ($S(O)$,P) the image of P under the mapping $S(O)$.

c) place (O,S)$\equiv$ sceneplace (stand(O),O,S), the point set occupied by O in scene S.

The sceneplace function may be used to describe the location of a subset or point in an object in a scene. Moreover, since a rigid mapping has a unique extension from O to the rest of R^3, sceneplace may also be used to locate a conceptual point outside the object which moves together with the object. For example, we can use it to name the position of the opening of a bottle or the position of the apex extrapolated from an object whose shape is the frustum of a cone.

Clearly, any possible position for an object is equally valid as the standard position. A change in the standard position of an object merely requires a change in the mappings associated

Axiom 3.1

For each objec

Definition 3.1

A mapping

M and a point set

We now define

scene is defined in

Definition 3.3:

A scene is

set of objects, O,

Some useful

Definition 3.4

Given a

mask(O),

a) extension

b) intension

c) plan

The

object in a scene

R^3

together with

R^3 to the

Given

change in the

with a scene.

Most physical reasoning involves histories, in the sense of [Hayes, 78]: chunks of space-time which can be isolated for contemplation. Because of the way we have defined scenes, our formal definition of histories differs slightly from his. We define a history as a continous function from time to scenes. (We identify time with the real line.) Thus, there might be a three minute history in which the robot paced back and forth in the room. The history would specify the position of all limbs and all other objects in the room at each instant of time in the three minute period.

Definition 3-5:

Given a set of objects U and an interval of time I either of form $[T_1, T_2]$ or of form $[T_1, inf)$, H is a history of U during I if H is a continuous function from I to scenes over U. (The topology on scenes is the obvious one.) If H is a history of U over I, we write formally, "history(H,U,I)", "$I=$ duration(H)", and "$U=$ objects(H)". For any time T in I, $H(T)$ designates the scene at time T.

A few obvious and useful definitions.

Definition 3-6:

scenes (H) is the set of scenes which occur in H.

Formally, scenes (H)= $\{ H(T) | T \in$ duration (H) $\}$

Definition 3-7:

H is *infinite* if its duration is infinite on the right.

Formally, infinite(H) iff exists (T_1)([T_1,inf)= duration (H))

Definition 3-8:

startscene(H) is the first scene in H.

Formally, startscene(H)=H (lowerbound (duration (H))).

Definition 3-9:

endscene(H) is the last scene of a finite history H.

Formally, if not (infinite (H)) then endscene(H)=H (upperbound (duration (H))).

Definition 3-10:

H is a *subhistory* of J if it is a restriction of J to a subset of the time interval.

Formally, subhistory(H,J) iff subset (duration (H), duration (J)) and forall ($T \in$ duration(H)) $H(T)=J(T)$.

Definition 3-11:

An *instantaneous* history is one which takes no time.

Formally, instantaneous(H) iff exists (T) $[T,T]$ = duration (H).

Definition 3-12:

hsequence(H_1,H_2) is the history consisting of H_1 and H_2 in sequence. If H_1 is infinite, then hsequence(H_1,H_2) is just H_1. Formally,

H= hsequence (H_1,H_2) iff exists (T_1,T_2,T_3)

[not infinite (H_1) and duration (H)=$[T_1,T_3]$ and

duration (H_1)=$[T_1,T_2]$ and duration (H_2) = $[T_2,T_3]$ and

forall (T)[($T_1 \leq T \leq T_2$) implies $H(T)=H_1(T)$] and

forall (T)[($T_2 \leq T \leq T_3$) implies $H(T)=H_2(T)$]] or

[infinite (H_1) and $H=H_1$]

Theorem 3-1:

If H=hsequence(H_1,H_2) then endscene(H_1)= startscene(H_2)

By abuse of notation, we use hsequence with more than two arguments to indicate sequencing of many histories.

Definition 3-13:

hsequence (H_1,H_2,$H_3 \cdots H_n$)= hsequence(H_1, hsequence(H_2, $\cdots$))

Definition 3-14:

initial (H_1, H) iff exists (H_2) [H = hsequence(H_1,H_2)]

H_1 is an initial part of H.

Definition 3-15:

endhistory (H_2,H) iff exists (H_1) [H = hsequence (H_1,H_2)]

H_1 is a final part of H.

The condition we placed on our physics, that all actions can be performed at an arbitrary rate, means that the time dependence of a history is essentially irrelevant. Two histories H_1 and H_2 can be considered equivalent if the scenes of one can be mapped monotonically into the scenes of the other. We do not allow stops to be inserted; thus, the relation between the histories is one-one.

Definition 3-16:

Histories H_1 and H_2 are *equivalent* under changes of time metric, written *hequiv*(H_1,H_2) iff

a) objects (H_1) = objects (H_2) = objects (H_3);

b) There exists a continuous, one-one, monotonically increasing function f from duration(H_1) to duration(H_2) such that forall ($T \epsilon$duration(H_1)) $H_1(T)=H_2(f(T))$

f) either H_1,H_2 are both finite or they are both infinite.

As long as we restrict attention to circumstances which respect the equivalence condition, we can replace most explicit references to time by uses of subhistory, initial, and hsequence.

The *trace* of an object through a history is the history restricted to that single object. The *motion* of an object through a history is its sequence of positions relative to the starting point. Thus, if H is the history of object O moving from $<1, 0, 0>$ to $<2, 0, 0>$ and object P rotating about its axis, then trace(O,H) is the history of O moving from $<1, 0, 0>$ to $<2, 0, 0>$, and motion(O,H) is the function "move one unit in the X direction".

Definition 3-17:

Definition 3.1...

initial $(H_i$ ...

H_i is an initial part ...

Definition 3.15:

satisfactory $(H_i(t)$ ...

H_i is a final part ...

The condition ...

rate, means that the ...

can be considered ...

the other. We do not ...

one.

Definition 3.16:

Historian H_i ...

a) objects $(h_i$ ...

b) There exists ...

to decrease ...

f) either H_i ...

As long as ...

can replace those ...

The ...

notion of an object ...

Thus, if ...

shown in ...

include ...

Definition ...

trace$(O,H)=I$ iff

objects$(I)=\{O\}$, duration$(I)=$duration(H), and

forall$(T\epsilon$ duration$(H))$ (scenemapping$(I(T),O)=$scenemapping$(H(T),O)$.

Definition 3-18:

motion$(O,H)=M$ iff

apply$(M(T)$,scenemapping(startscene$(H),O))=$scenemapping$(H(T),O)$

Finally, we fix a notational convention for convenience. Most of our geometric terms are defined on bodies, but are applied to objects in a scene. So strictly, we would continually find ourselves writing

$$geometric\_term(place(O_1,S),place(O_2,S) \cdots place(O_n,S))$$

We will abbreviate all such expressions as $geometric\_term(O_1,O_2 \cdots O_n,S)$. For example, if we have defined abut(B_1,B_2) to mean that bodies B_1 and B_2 have intersecting boundaries, then we can write abut(O_1,O_2,S) to mean that the places of objects O_1 and O_2 abut in scene S.

4. Constraints

The next problem is to represent constraints. So far, a scene can be any set of mappings at all, and a history can be any continuous function onto scenes. We must have a mechanism for ruling out many of these as physically impossible. We use *constraints* to define which histories are consistent with physical laws. Constraints can also be used to limit histories in ways more specific than the general physical laws; in particular, to express physical constraints associated with specific situations, and restrictions imposed on the problem solver. An example of the former would be a joint connecting two limbs, which restricts their relative positions in a particular way. Here the constraint rules out histories which take the objects into excluded positions. An example of an imposed restriction would be a rule that the robot cannot allow any object to drop. Here the constraint rule out any history where any object drops.

Formally, a constraint is a set of histories; those which satisfy the constraint. We note that if a history occurs then all its subhistories occur and hence must also satisfy the constraint. We also

require that our constraints be closed under temporal equivalence. Finally, we require that constraints be have no memory; that is, whether a sequence of events is allowed by a constraint depends only on the state of the world during the events, and not on previous history. This can be stated formally as the requirement that if two histories satisfy the constraint, and they can be concatenated then their concatenation is in the constraint.

Definition 4-1:

A *constraint* C is a set of histories such that

a) if H is in C and I is a subhistory of H then I is in C;

b) if H is in C and hequiv (I,H) then I is in C.

c) if H is in C and I is in C and endscene (H) = startscene (I) then hsequence (H,I) is in C.

A number of specific constraints are particularly significant. These include the constraint that hard objects don't overlap; the constraint that only animate objects are capable of spontaneous motion (a law of naive physics, at least in some contexts, though not a law of "real" physics, and an approximation often made in robot planning); and the law of gravity. We may formally define the constraint on hard objects as a set of histories as follows: The most basic constraint is the discreteness of hard objects: hard objects do not overlap. This constraint is called *hardobjs* and may be defined as follows:

Definition 4-2:

hardobjs $\equiv$ $\{ H \mid$ forall $(O_1, O_2 \in$ objects(H), $S \in$ scenes$(H))$ overlap(O_1, O_2, S) implies $O_1 = O_2 \}$

Overlap, used above means that two bodies have overlapping interiors

Definition 4-3: Given two bodies B_1 and B_2, overlap(B_1,B_2) iff interior(B_1) intersects interior(B_2), where interior is taken in the strict topological sense. (The interior of a box is inside the cardboard, not in the air surrounded by the box.)

We defer the formal definitions of the constraint that inanimate objects do not move spontaneously, and the law of gravity to section 9 on account of their complexity.

Physical constraints occupy a different logical status in planning than non-physical contraints. The robot must work to maintain imposed constraints, while physical constraints maintain themselves. This distinction becomes important in defining feasibility of actions (see section 6).

We may treat a set of physical constraints as a single constraint which is the intersection of the various constraints in the set. We will use this abbreviation to simplify the exposition below.

5. Senses

We presume that the robot is equipped with sensors which, under specified conditions, make certain measurements. Measurements may be other than real valued; for example, a boolean valued measurement corresponds to verifying a fact. Sensors are of two kinds. There are scene-sensitive sensors, which measure quantities in a scene; and there are history-sensitive sensors, which measure quantities in a history. For example, a gasoline gauge is a scene-sensitive sensor while a speedometer or an odometer is a history-sensitive sensor. In this paper, we will deal exclusively with scene-sensitive sensors.

Abstractly, a sensor is a function on scenes. We will allow the value of a sensor function to be any type of mathematical object. All sensors can sometimes return the value "undefined", in cases where the sensing cannot be performed. Given a sensor Q and a scene S, we say that Q is detectable in S if $Q(S) \neq$ undefined.

For example, let Q be an electric eye with source X and receptor Y (both source and receptor assumed to be point objects). $Q(S)$ is the boolean predicate "there is some object which intersect the line segment <sceneplace(X,Q,S) -- sceneplace(Y,Q,S) >". (Note that we are here confounding the sensor Q with the physical object with which it is associated.) For another example, if Q is a two-dimensional camera eye in a world of unknown black objects against a white background, then $Q(S)$ is the projection of the objects in the world against the plane of Q, a subset of R^2. As the camera becomes more sophisticated, the world becomes more varied, and more prior knowledge is available, the value of $Q(S)$ becomes more complex.

As with constraints, a set of sensors $Q_1, \cdots Q_n$ can often be considered as a single sensor Q

which returns all of the results of the separate sensors combined into a tuple: $Q(S)=<Q_1(S),Q_2(S) \cdots Q_n(S)>$.

A robot is defined as a class of sensors plus a class of objects which constitute its body.

6. Actions

An action is something that an actor can do in a given situation to make things happen. An actor can bring it about that certain histories occur and others do not. However, his control over the universe is limited; he can cause certain things to happen, but other parts of the universe may pursue activities independent of him. Therefore, the result of an action in a particular scene is a set of histories compatible with that action. We may, in fact, identify the action with all the histories which are compatible with that action in any starting scene. The result of an action in a given scene is simply that subset which begins with the particular scene.

This definition is somewhat non-intuitive, so it is worth elaborating. We might naturally reason that, when you perform an action in a scene, some specific thing happens, which would suggest a definition of action as a function which takes a scene into a history. This is not acceptable, for several reasons. We note that the resulting history depends, not only on the scene but also on the active physical constraints. Quite different things will result from the action of trying to lift the handle of a suitcase in the cases where (a) the handle is not attached to the suitcase, (b) the handle is attached, and (c) the handle is attached to the suitcase, but the suitcase is glued to the ground. We might then modify our definition of an action as a function which takes a scene and a set of constraints as input, and a history as output. But the function is of a very well behaved kind; the history must begin with the scene, and must be an element of the set of constraints. This suggests defining the action as simply the union of all the possible results under all possible sets of constraints. We can then define the effects of starting in a scene or obeying physical constraints as restrictions on that set.

Definition 6-1:

An action is a set of histories.

Definition 6-2:

Given an action A and a scene S, results(A,S) $\equiv$ { H | $H \epsilon A$ and startscene(H)=S } (read "the results of A in S")

Note that this definition of action is more general than the state-space definition of action as a function from states to states, in that it allows "run around the track three times" as a possible action. It is less general, but rather more concrete, than the analysis of action of [McDermott, 82], in which "action" is allowed to include preventing, thinking, noting, etc.

We allow impossible actions. We allow the action "lower the book 3 feet" even if the book is lying on the desk, and the desk is fixed. The result is simply a history of the book going through the desk. This is impossible but not undefined. Our definition of an action being "feasible" below captures this sense of impossibility. Meanwhile, it simplifies many definitions if we define actions as broadly as possible.

Definition 6-3:

A *situation* is an ordered pair of a scene and a physical constraint.

Definition 6-4:

The *possible results* of an action A in a situation $<S,C>$,

posresults(A,S,C)$\equiv$$\{H | H \epsilon A \cap C$ and startscene(H)=S $\}$

The feasibility of an action depends on the starting scene, the physical constraints, the available sensors, and the class of objects that you control directly. Actions which violate the physical constraints are said to be physically infeasible. Those which require information undetectable by the sensors are called epistemically infeasible.

Physical feasibility depends only on the extension of the action -- that is, on the set of histories which is the action -- and on the situation, the combination of the starting scene and the assumed physical constraints. An action is physically feasible in a given situation only if there are some histories in the action which begin with the scene and which obey the constraints. The assumption is that the world will accommodate your action if it is in any way compatible with the

Definition 5

Given an actio ... Post and action $W(p)$ ...

results of A in S'.

Note that this d ... that the ...

a function from states ... that ...

action. It is less general ...

in which "action" is allowed ...

We allow impossible actions ...

lying on the desk, and ...

the desk. This is impossi ...

captures the sense ... we ...

as broadly as possible.

Definition 5.

A situation ...

Definition ...

The postul ...

postcond(A, S) ...

The feasibility ...

this reason, and ...

constraints are ...

the effects are ...

Physical ...

terms which ...

essential physica ...

these theories ...

assumptions ...

constraints.

However, we do not want to force the world to be prophetic. The world has no way of forseeing the end of your action at its beginning; it can only be accommodating locally. Therefore, in a condition where you and the world get into a dead end -- you pursue your action for a while, the world does something compatible for a while, and then you find that the rest of the action is incompatible with the current state of the world -- the original action must be judged infeasible, even if there is something else the world could have done which would have allowed your action to go through. We therefore impose the additional condition that any successful beginning of the action be extendible to a successful completion.

Definition 6-5:

An action A is *physically feasible* in a situation $<S,C>$ iff

1) posresults(A,S,C) is non-null; and

2) for each history H in results(A,S), if I is an initial part of H and I is in C then there exists a J in posresults(A,S,C) such that I is an initial part of J.

An action is epistemically feasible with respect to a sensor Q (representing, perhaps a set of sensors) if the robot can use Q to keep his motions within the action. We will first define the special case of a *blind* action, an action which can be executed without any sensors whatever. Such an action is characterized by the fact that the motion of the robot is entirely unaffected by any other object.

Definition 6-6:

An action A of robot R is *blind* iff

forall $(H_1 \epsilon A, H_2)$ [motion(R,H_1)=motion(R,H_2) implies $H_2 \epsilon A$].

(Strictly speaking, since the robot may have multiple body parts, we should quantify over objects which are part of the robot. However, we will write our definitions as if the robot was a single object for simplicity.)

Thus whether a history is in A is determined wholly by the motion of the robot in R. Note

that this property is only true of the action as a whole; it does not apply when the action is restricted by physical constraints. For example, if the action is "move two feet forward" and the constraint that hard objects do not overlap applies then the motion of other objects is not independant of the motion of the robot; if they are in its way, it will push them. But considering performances of the action under all possible physical laws, the action does include histories where the objects stand still, and the robot walks through them.

To include sensors, we must allow the robot to adjust his motions according to what he detects in the outside world. We therefore weaken the above definition as follows. If a given motion is permitted in one set of circumstances and not in another, then the difference between the circumstances must be detectable by the sensors. Moreover, a detectable difference must arise no later than the forbidden motion "becomes" forbidden. We can formalize this as follows:

Definition 6-7:

An action A by robot R is epistemically possible under sensor Q iff

forall $(H_1 \epsilon A, H_2)$

[[not $(H_2 \epsilon A)$ and motion(R, H_1)=motion(R, H_2)] implies

exist (J_1, J_2, I)

[initial(J_1, H_1) and initial(J_2, H_2) and initial(J_2, I) and $I \epsilon A$ and

motion(R, J_1)=motion(R, J_2) and Q(endscene(J_1)) $\neq$ Q(endscene(J_2))]]

For example, let Q be an action which can detect the position of object A in some, but not all positions. Then the action "if $Q(S)$ is not undefined, move halfway toward A; else, stand still" is epistemically possible. If the motion of the robot ends up different in two histories, it is either because A was detectable in one and not the other, or because A was detectable in both and had different positions relative to the robot. The unqualified action "move halfway toward A" is, in general, epistemically infeasible, because there will be two starting scenes where the position of A is different but undetactable. The required motion in these two scenes must be different, and the sensor is not adequate to guide this difference.

It is interesting to contrast this treatment of epistemic feasibility with that in [Moore, 80]. We are obviously more restricted in domain, but our more concrete approach allows us to describe the whys and wherefores of feasibility in cases where Moore simply adopts *ad hoc* axioms. For example, Moore uses axioms which assert that a given action produces a given piece of knowledge; he does not give a mechanism to explain why an action would be associated with the knowledge. Similarly, it is an axiom in Moore's system that the action of opening a safe is rigidly designated by the specification "turn right to 80 then left to 40 then right to 50" but not by the specification "turn right to the first number of the combination, then left to the second, then right to the third". He uses this axiom to prove that one must know the combination before opening it. This level of abstraction gives his system generality, but requires a lower level of analysis to rest on.

Our results provide such support in our domain. In the vocabulary we have developed, we can state axioms about safes which would allow us to prove such result as "there is no epistemically possible action which will always open a safe of unspecified combination in 3 turns," and "given any safe of specified combination, there is an epistemically possible action which will always open it". The gap here our concept of "unspecified" and Moore's concept of "unknown" is easily bridged. Moreover, we are in a position to prove theorems of a kind difficult in Moore's system such as "If a blinking light turns on whenever the correct number of the combination is reached, then the safe can be opened even without prior knowledge of the combination." It might be worthwhile to try to combine the two definitions into a single coherent system.

One final property of actions is determinism. An action is deterministic if, given any fixed behavior of the surrounding objects, there is only one possible behavior for the robot. We first define two histories H and I as distinct if neither H nor I is equivalent to any subhistory of the other. We can now define a non-deterministic action as follows:

Definition 6-9:

An action A of robot R is non-deterministic iff

It is interesting [illegible]
We are obviously more [illegible] the same message [illegible]
the ways and [illegible] to [illegible]
example, [illegible]
he does not give a [illegible]
Similarly, it is no [illegible]
by the specification "[illegible]"
"turn right to the [illegible]"
The user has [illegible]
elaboration gives [illegible]

Our results [illegible]
can state axioms about [illegible]
only possible [illegible]
"given any set of [illegible]
upon it". The [illegible]
integral. Moreover [illegible]
such as "If a [illegible]
then the only can [illegible]
worthwhile to [illegible]

The final [illegible]
behavior of the [illegible]
define two histories [illegible]
other. We can now define [illegible]

Definition [illegible]

[illegible]

exist($H_1 \epsilon A, H_2 \epsilon A, I, J_1, J_2$)

initial(J_1,H_1) and initial(J_2,H_2) and initial(I,J_1) and initial(I,J_2) and

distinct(trace(R,J_1),trace(R,J_2) and

forall(O) ($O \neq R$) implies trace(O,J_1)=trace(O,J_2))

That is H_1 and H_2 start out the same in I; everything else stays the same through J_1 and J_2, but R does different things.

7. Objectives and Tasks

In the most general terms, an objective for a robot is to force some class of histories to occur. This could include such objectives as "Run around the block three times", "Get to first base before the ball", or "Swing the child". Very frequently, the objective histories are distinguished as ending with a scene of a particular type; it doesn't matter how they go as long as they end up in the right place. Such objectives include "place the book on the shelf", "attach the nut to the bolt", and "get away from me".

Often such desired classes of scene are important because they restrict the class of histories which can begin with them. Three very common functionalities are of this kind. "holding(O,H)", where O is an object and H is a class of histories, is the class of scenes where O is held against actions in H; that is, all histories in H starting with a holding scene are physically impossible. For example, a book on a table is held against histories beginning with a downward motion (such as falling histories). "fasten(O,P,HH)" means that O is fastened to P, barring histories of type HH (unfastening histories); that is, all subsequent histories must either be of type HH or must have O and P within some maximum distance. "protect(O,U,HH)" means that O is protected against objects in class U performing histories of type HH; that is, no history of type HH starting in a protected scene will cause objects in U to abut object O. For example, a scene where you have an umbrella over your head is an element of protect (you, raindrops, histories where raindrop move downward and you do not lower the umbrella).

A task for a robot consists of an objective, a set of active physical constraints, a class of starting scenes (the robot knows he is in one, but he does not know which), and set of imposed

constraints to be observed. (Recall that the robot himself consists of sensors and body.) A solution for the task is an action which is feasible in any of the starting scenes and all of whose results from any of the starting scenes lies within the objective. This is the motion planning problem. It includes such problems as "Put the book somewhere off the floor without knocking over the vase".

A design problem starts with an objective, a set of physical constraints, a starting scene and a subspace of the scene representing accessible space. A solution for a design problem consists of a new object, a starting place somewhere in accessible space, and a plan such that, when the new object is added at the place in the scene, the plan will be feasible and will achieve the objective. For example, we might be presented with the objective of supporting a coffee cup off the floor in a particular corner of the room. A solution would be a side table of the specified size and shape, a starting place somewhere in the room, and a plan to put the table in the corner and to put the cup on the table.

8. Net Motions

In the remainder of this paper we will use the semantics developed to state some of the "natural laws" of naive physics in terms of physical constraints. We begin by borrowing a number of ideas and terms from linear algebra to describe rigid motions of objects. We must first define the two basic motions, translate and rotate.

Definition 8-1: For any vector X, translate(X) is the function mapping a vector U to $U+X$. In lambda notation, translate(X)$=\lambda(U)(U+X)$.

To define rotation, we need some preliminary definitions:

Definition 8-2: A *direction* is a unit vector.

Definition 8-3: A *directed line* L is an equivalence class of pairs $<x,u>$ where x is a vector and u is a direction, and where $<x,u>$ is equivalent to $<y,v>$ iff $u=v$ and $y=x+tu$ where t is a real number. If $<x,u>\in L$, we write $L=$line(x,u).

Definition 8-4: $L=$line(x,u) and $M=$line(y,v) are *parallel* iff $u=v$ and *anti−parallel* if $u=-v$

conditions to be observed [illegible] that the [illegible] (essentially) depend [illegible]

for the task is an action [illegible] and [illegible]

from any of the starting [illegible] This is the answer (ignoring [illegible]

includes such positions [illegible] the [illegible] without knowing [illegible]

A design problem [illegible] of constraints [illegible]

a subspace of the state [illegible]

new object, a starting [illegible] than that [illegible]

object is added to the [illegible]

For example, we might [illegible]

a particular [illegible]

starting place [illegible]

top on the table [illegible]

8. The [illegible]

In the [illegible]

"natural laws" of [illegible]

of ideas and [illegible]

the two kinds of [illegible]

Definition [illegible]

In contexts [illegible]

Definition [illegible]

Definition [illegible]

Definition [illegible]

and a [illegible]

not [illegible]

Definition [illegible]

Definition 8-5:

Given a line $L=\text{line}(x,u)$ and a vector Y, the *projection* of Y onto L, $\text{lproj}(Y,L)=x+(Y\cdot u)u$.

The *perpendicular* from Y to L, $\text{perp}(Y,L)=Y-\text{lproj}(Y,L)$.

The *unit perpendicular*, $\text{uperp}(Y,L)=\dfrac{perp(Y,L)}{||perp(Y,L)||}$.

The *normal* of Y and L, $\text{norm}(Y,L)=$ cross-prod$(u,\text{uperp}(Y,L))$.

The *unit normal*, $\text{unorm}(Y,L)=\dfrac{norm(Y,L)}{||norm(Y,L)||}$.

Definition 8-6: Given a line L and an angle θ,

$$\text{rotate}(L,\theta)=\lambda(Y)\ (\text{lproj}(Y,L)+||\text{perp}(Y,L)||\cdot(\cos(\theta)\ \text{uperp}(Y,L)\ +\ \sin(\theta)\text{unorm}(Y,L))$$

Given two rigid motions R and S, $S\cdot R=\lambda(X)(S(R(X)))$. We follow the analysts' convention of writing the functions in backwards order of the order of application order. The inverse of a rigid motion R, R^{-1} is the motion such that $R\cdot R^{-1}=I$, the identity motion.

The rigid motion which relates the positions of an object in two different scene is called the net motion from one scene to the other.

Definition 8-7:

Given an object O and two scenes S_1 and S_2,

net-motion$(O,S_1,S_2)=$scenemapping$(O,S_2)\cdot$ (scenemapping$(O,S_1))^{-1}$

Given object O and history H,

hnet-motion$(O,H)=$net-motion $(O,$startscene (H),endscene$(H))$.

It is a basic theorem of linear algebra that any rigid motion can be written as the composition of a rotation and a translation.

Theorem 8-1: Given any rigid motion R, there exist X, L, and θ such that $R=\text{translate}(X)\cdot\text{rotate}(L,\theta)$. This is called a *rotation-translation decomposition* of R. We write $R=\text{rot-trans}(X,L,\theta)$. Two decompositions of the same motion, $R=\text{rot-trans}(A,L,\theta)=\text{rot-trans}(B,M,\omega)$, satisfy the following relations:

1) Either parallel(L,M) and $\theta = \omega \bmod 2\pi$ or anti-parallel(L,M) and $\theta = -\omega \bmod 2\pi$

2) lproj(A,L)=lproj(B,M)

It is also the case that given any line satisfying (1) above, or any vector satisfying (2), it is possible to find a decomposition of the motion using that line or that vector (unless $\theta=0$). It is often useful to restrict the range of decompositions being considered to a more well-behaved class. Generally we do this by restricting the range of axes of rotation. Sometimes it will be convenient to oblige the axis of rotation to pass through a specified point, such as the center of mass. Such a constraint imposes a unique value on the decomposition (except that the direction of the line may be reversed). A more flexible constraint is that the line of rotation pass through some point in the convex hull of the object. Such a decomposition is called *internal*. A formal definition is as follows:

Definition 8-8:

Given an object O and two scenes S_1 and S_2, a triple <A,L,θ> is an *internal decomposition* iff

1) rot-trans(A,L,θ) = net-motion(O,S_1,S_2); and

2) L intersects the convex hull of place(O,S_1).

Definition 8-9: Given an object O and two scenes S_1 and S_2, the *internal translation space* of the motion of O is the set of vectors A for which there exists L and θ such that <A,L,θ> is an internal decomposition of the motion of O.

If we retrict the axis of revolution to the center of mass then we have a unique decomposition. The translational part of this decomposition is simply the displacement of the center of mass between the two scenes. Given O, S_1, and S_2, we designate this decomposition $cm-motion(O,S_1,S_2)$ and we designate its translational part $cm-translate(O,S_1,S_2)$. Likewise, abusing notation, we write cm-motion(O,H) and cm-translate(O,H) in referring to motion from the starting to the ending scenes of a history H.

1) Either parallel(L,M) ...

2) ...(L,C) = ...

It is also the case ...

possible to find a ...

often useful to restrict ...

Generally we do this by ...

to design the axis of rotation ...

reasonable ...

be ...). A ...

convex hull of the ...

lines:

Definition ...

Given an object ...

iff

1) ...

2) A ...

Definition ...

the ...

internal ...

If we ...

then. The ...

between the ...

...

...

the ...

9. More Physical Constraints

In this section we will investigate, in greater depth, typical physical constraints which arise in our domain. We have already discussed the "hard objects" constraint in section 4. Another common constraint is that only animate creatures (in which we include robots) are capable of spontaneous movement; other objects move only when the robot is in direct or indirect contact with them. This constraint is of course not true even in the most naive of physics, as it ignores not only inertia but also falling. However, it is an approximation which is often useful. We can express this constraint as follows:

Definition 9-1:

animate_driven=

{ H | forall(O) [movesin(O,H) implies

exists ($S \in$scenes(H),R)

[animate(R) and indirect_touch(R,O,objects(S),S)]] }

What this definition actually states is that if O moves in H then R is in indirect contact with O at some point within H. Since histories are continuous, it follows that R is indirect contact with O whenever O is moving.

Definition 9-1 relies on three undefined terms: "movesin", "animate", and "indirect_touch". "Animate" is of course undefinable within our theory; it is simply a predicate which applies to some objects, or sets of objects, and not others. The other two may be defined as follows:

Definition 9-2:

If O is an object and H is a history then

movesin(O,H) iff exist (S_1,$S_2 \in$ scenes(H)) [scenemapping(O,S_1)$\neq$scenemapping(O,S_2)]

Definition 9-3:

If P and Q are bodies, and S is a set of bodies, then

indirect_touch(P,Q,S) iff exists ($B_1=P$,$B_{2,} \cdots B_n=Q$) [abuts (B_i,B_{i+1},i=1 to n)]

Definition 9-4:

If P and Q are bodies then abuts(P,Q) iff the boundaries of P and Q intersect, and their interiors are disjoint.

The constraint "animate_driven" is quite weak. For example, it allows a robot to slide an object along his body, without moving any of his body parts, or, even stranger, to touch one end of a row of blocks, and move the block at the far end without moving any of the blocks in the middle. A more reasonable constraint would be that the motion must involve a chain of objects, starting with the robot, where each object pushes on the next. We call this constraint "push_driven" and define it as follows.

Definition 9-5:

push_driven ≡

{ H | forall (O) movesin(O,H) implies

exist(H',R) [subhistory(H',H) and animate(R) and pushing(R,O,H')]}

Pushing(R,O,H') means that R was pushing O, directly or indirectly, throughout the subhistory H'. We define *pushing* in terms of chains of "direct push" as follows:

Definition 9-6:

pushing(P,O,H) iff $P=O$ or exist $(O_1=P, O_2, \cdots O_n=O)$ [direct_push (O_i, O_{i+1}, H)]

Object A directly pushes B throughout H if in any sufficiently small subhistory, A is moving into B's space. Formally

Definition 9-7:

direct_push(A,B,H) iff

forall (H_1) [subhistory(H_1,H) implies

exists (H_2) [subhistory(H_2,H) and

overlap(place(A,endscene(H_2)),place(B,startscene(H_2))]]

Together with the "hard_objects" constraint, the "push_driven" constraint guarantees that objects will move if and only if the something pushes on them, and that push originates with animate objects. The direction in which the pushed object moves, however, is left open, as long as it

moves out of the way of the pusher. A complete analysis of the response to pushing is probably impossible without a Newtonian force analysis, together with a theory of friction.

In many cases, we must supplement the simple theory given above by a simple theory of gravity. We do not wish to import the theory of gravity as universal attraction, or even as a uniform constant force field. Rather, we are only interested in the interaction of gravity with controlled motions. We cannot ignore falling altogether -- something has to happen when you push a vase off a table -- but we treat it as an arbitrary motion with a downward component. For our purposes, gravity imposes two constraints on histories, one negative and the other positive. Firstly, no object moves unless one of the following is true: the object is animate; the object is pushed; the object is carried; or the motion is downward.

Secondly, all objects move directly downward if it is not supported. A supported object may move downward if it can push the support out of the way, or if it can roll out of the way.

Note that this account ignores inertia, both upward and horizontal. The second constraint disallows both throwing an object up, and having an object fly off a surface in a horizontal direction, because it would be an unsupported object which is not moving directly downward. In fact, there is no way to introduce inertia into the system, and still disallow objects from spontaneously starting to move upward, without either adding velocity as a state variable, or making constraints history dependent. The physics described by the above constraint are basically those of slowly moving objects in a viscous medium which absorbs any inertia -- snails pushing blocks in a sea of liquid Prell.

We define downward motion in terms of the center of mass. To simplify our theory, and to allow it to encompass cases where the distribution of mass is unknown, we assume about the center of mass only that it is some point within the convex hull of each object.

There is a difficulty in defining carrying. What we would like to say, intuitively, is that, if object O supports object P, and object O moves, then P will move along with it, if it possibly can. However, this formulation is non-monotonic, since it relies on the non-existence of a proof that P will be prevented; and non-monotonic axioms are, in general, to be avoided. There are two ways

of stating the axioms monotonically: either that P must move along with it, or that P may move along with it. The first formulation is definitely too strong. If P must move with O, then O cannot move without P. From this we could deduce that even if some other support caught hold of P still O could not get away from P. We must, therefore, adopt the second formulation and live with the fact that our physics does not give us any assurance that objects we have placed on a horizontal surface will stay there when the surface is moved.

We need five primitives to define these constraints formally. The first is *up*, which is a direction. We will associate z with the up direction, and x and y with the horizontal directions. The second is the *ground*. The ground is an immobile object which extend indefinitely far down from its surface. Axiom 9-1 expresses the latter condition.

Axiom 9-1:

forall(x,y) exists(z_0) forall ($z<z_0$) $<x,y,z> \epsilon$ stand(ground)

Up and *ground* are constants of the theory. *Center-of-mass* is a function from objects to a point in R^3. We locate the center of mass with respect to the standard position of the object. It satisfies the following axiom:

Axiom 9-2: forall(O) center-of-mass(O) ϵ convex-hull(stand(O)).

Finally, we need two support predicates: *direct_support*(B,C), meaning that body B directly supports body C; and *ground_supported*(O,S), meaning that O is ultimately supported by the ground in scene S. These are defined as follows:

Definition 9-8:

B directly supports C iff there exists a vertical line L and a point p on L such that p is on the boundaries of both B and C, and there exists a small open interval of L bounded below by p which is in the interior of C, and there exists a small open interval of L bounded above by p which is in the interior of B.

This is clearly a very weak definition of "direct-support". It allows B to support C even if they meet only at a single point, or even if they meet along a very steep face. However, the con-

straints above which involve support are also quite weak. It may be noted that the idea of a stable support, which holds the center of mass in a local minimum, is built into the gravity constraints. The main weakness of our definition of direct support is that it does not distinguish between supports which are labile and those which are completely unstable, but this is a somewhat subtle distinction for many purposes.

Now, we can define "ground_supported", as the transitive closure of "support", terminated by the ground.

Definition 9-9:

ground_supported(O,S) iff exist $(O_1=ground, O_2, \cdots O_n=O)$ such that *direct_support*(O_i, O_{i+1}, S)

We are now in a position to define the gravity constraint. We will piece it together out of a number of separate constraints. *Stationary_ground* states that the ground stands still.

Definition 9-10:

stationary-ground $\equiv$ { H | ground ϵ objects (H) and forall ($S \epsilon$ scenes(H)) scenemapping(gound,S)=I, where I is the identity mapping.

Must_fall is the constraint that unsupported objects fall.

Definition 9-11:

must_fall $\equiv$

{ H | forall ($O \epsilon$objects(H), H_1)

[[subhistory(H_1,H) and not instantaneous (H_1)] implies

[ground-supported(O,startscene(H_1)) or

exists (H_2) [initial(H_2,H_1) and falls(O,H_2)]] }

Must_fall uses falls(O,H) as a predicate meaning object O during history H.

Definition 9-12:

falls(O,H) iff exists(k>0) cm-translate(O,H)=$-kz$.

We now state the constraint that no object move upwards spontaneously.

Definition 9-13:

no_up ≡

{ H | forall($O \epsilon$ objects(H))

movesin(O,H) implies

exist(H_1,R) [subhistory(H_1,H) and drives(R,O,H) and animate(R) or falls (R, H)]] }

"Drives" is defined analogously to "pushing" above. P drives $\$O$ if there is a chain of objects from P to O such that each either pushes or carries the next.

Definition 9-14:

drives(P,O,H) iff $P=O$ or

exist ($O_1=P, O_2, \cdots O_n=O$)

[direct_push(O_i,O_{i+1},H) or direct_carry(O_i,O_{i+1},H)]

Definition 9-15:

direct_carry(O,P,H) iff forall($S \epsilon$ scenes(H)) direct_support(O,P,S)

Finally we can define the gravity constraint:

Definition 9-16:

gravity ≡ hardobjs $\bigcap$ stationary_ground $\bigcap$ must_fall $\bigcap$ no_up

10. Conclusions

The system presented develops an integrated set of axioms for the following domains:

1) The naive physics of rigid objects, in cases where neither time nor force has to be quantified;

2) Physical actions, physical feasibility, and action descriptions;

3) Sensors and epistemic feasibility

For many important problems in robotics and naive physics, such as those listed in section 1, these domains will suffice. We can formally describe these problems in the terms we have

developed. Papers in preparation will describe a robot programming language based around these concepts and a study of the geometrical primitives needed in this domain.

Acknowledgements

Thanks to Ralph Grishman, Malcolm Harrison, Drew McDermott, Leora Morgenstern, Colm O'Dunlaing, Ed Schonberg, Jack Schwartz, and Paul Spirakis for their helpful criticisms.

Bibliography

Brooks, Rodney "Solving the Find-Path Problem by Good Representation of Free Space", AAAI-82 p. 381 - 386, 1982

DeKleer, Johann and John Seely Brown "Assumptions and Ambiguities in Mechanistic Mental Models" in Gentner and Stevens, (ed.) *Mental Models*, Lawrence Erlbaum Associates, 1983

Forbus, Kenneth "Qualitative Reasoning about Space and Motion" in Gentner and Stevens, (ed.) *Mental Models*, Lawrence Erlbaum Associates, 1983

Hayes, Patrick "The Naive Physics Manifesto" in D. Michie (ed.) *Expert Systems in the Micro-Electronic Age* Edinburgh University Press, 1979

Kuipers, Benjamin "Commonsense Reasoning About Causality: Deriving Behavior From Structure", Tufts University Working Paper in Cognitive Science #18, May 1982

Lozano-Perez, Tomas "Automatic Planning of Manipulator Transfer Movements" IEEE Transactions on Systems, Man, and Cybernetics, SMC-11, p. 681 - 698, 1981

McDermott, Drew V. "A Temporal Logic for Reasoning About Processes and Plans", *Cognitive Science* Vol 6, pp. 101 - 155 1982

McDermott, Drew V. "Reasoning about Plans" in Jerry Hobbs (ed.) *Formal Theories of the Common Sense World* to appear

Moore, Robert, *Reasoning about Knowledge and Action*, SRI AI Center, Technical Report 191, 1980

Sacerdoti, Earl, *A Structure for Plans and Behavior*, American Elsevier Publishing Co. 1977

Schwartz, Jacob and Micha Sharir "On the 'Piano Movers' Problem I. The case of a Two-Dimensional Rigid Polygonal Body Moving Amidst Polygonal Barriers", Tech Report 39, Computer Science Dept., Courant Institute, 1981

Wilensky, Robert "Metaplanning" Technical Memo #33, Berkeley Dept. of Computer Science, 1980